Metric for Security Activities assisted by Argumentative Logic

Tarek Bouyahia, Muhammad Sabir Idrees, Nora Cuppens-Boulahia, Frédéric Cuppens, Fabien Autrel

Metric for Security Activities assisted by Argumentative Logic

Tarek Bouyahia, Muhammad Sabir Idrees, Nora Cuppens-Boulahia, Frédéric Cuppens, and Fabien Autrel

Telecom Bretagne, 2 rue de la châtaigneraie, Cesson Sévigné, France

Abstract. Recent security concerns related to future embedded systems make enforcement of security requirements one of the most critical phases when designing such systems. This paper introduces an approach for efficient enforcement of security requirements based on argumentative logic, especially reasoning about activation or deactivation of different security mechanisms under certain functional and non-functional requirements. In this paper, the argumentative logic is used to reason about the rationale behind dynamic enforcement of security policies.

Keywords: Argumentative logic, Reasoning, Complex attack, Reaction policies.

1 Introduction

Designing a secure system has always been a complex exercise. In practice, much of the focus for designers and developers being on delivering a working system in the first place; on the other hand, security concerns have long been considered only in retrospect, especially after serious flaws are discovered. Security experts are thus generally confronted with an existing system, whose architecture might actually hamper the deployment of security mechanisms that would prevent the occurrence of the attacks they envision. On the contrary, the challenges of modern security tools is to keep the system in a safe state while maintaining the best possible level of performance and quality of service. From the embedded system viewpoint, enforcement of security requirements becomes even more challenging and more critical. These challenges stem from the tight relationship between architecture design and its functional, and non-functional requirements as well as their impact on one another. For instance, if the system architecture design changes or evolves, these requirements should meet the new architecture design objectives and choose the best countermeasure that can be applied in this specific context or situation. This is especially true in safety-critical systems such as automotive systems [6,13], where attacks may be devastating, but where security functions overhead may also result in an absolutely useless system. In such a context, designing a secure system has always been a complex exercise. Indeed, security is a functionality that is difficult to specify and implement because it is

not modular: modifications to one part of an application may interact strongly with the security properties of other parts of the same application.
In this paper, we present an approach that solves these problems. This approach is driven by argumentative logic (AL)[8]. It describes a structured collaboration and interrelationship between the system architecture design and security requirements to support the long-term needs of the system. The purpose of security activities assisted by argumentative logic is to bring into focus the key areas of concern, highlighting the decision criteria and security context for each system aspect that has direct or indirect value for a stakeholder. We also claim that the security analysis should also play an important role with respect to convincing the designer of increasingly complex embedded systems of the consistency and exhaustivity of his reasoning and selection of security measures, at least with respect to the identified threats. The use of argumentative logic driven reasoning engine can help in dynamic enforcement of security mechanisms through the introduction of non-monotonic reasoning capabilities. This capability opens up the door to the dynamic selection and enforcement of security mechanisms performed statically only today.
The paper is organized as follows: Section 2 introduces a case study from an automotive domain we use throughout the paper. Section 3 goes around works already done within the scope of this paper. Section 4 explains our approach for dynamic enforcement of security requirements assisted by argumentative logic. We show deployment scenario highlighting how an argumentative logic driven reasoning engine, which makes it easier to dynamically enforce security mechanisms in Section 5. Section 6 concludes the paper and outlines future work.

2 Motivation example for Efficient Security Enforcement

In order to give an example of potential need for dynamic enforcement of security requirements to control different security activities, we consider the following abstract example of automotive on-board system. A modern automotive on-board network interconnects a hundred of microcontrollers, termed Electronic Control Units (ECUs) organized into application-specific domains bridge by gateways, as shown in Figure 1. Attacks have been shown to be quite feasible [9] by bypassing the filtering performed between domains or by brute-forcing ECU cryptography-based protection mechanisms. Such attacks may in practice originate from the Internet connection increasingly available in vehicles or even from the Bluetooth pairing of a compromised mobile phone to the vehicle on-board network. Further attacks are anticipated in upcoming Car2X applications, which will feature vehicle-to-vehicle or vehicle-to-infrastructure communications. Many security attacks and vulnerabilities are due to the fact that either security policy is not well specified and enforced or system-wide security policies (dependencies between different security policies) are too weak. Automotive on-board architectures do not only rely on the simple enforcement of security rules but also involve multiple enforcement points, especially when the underlying platforms and infrastructures are providing services themselves, like HSM, or middleware layers. For instance,

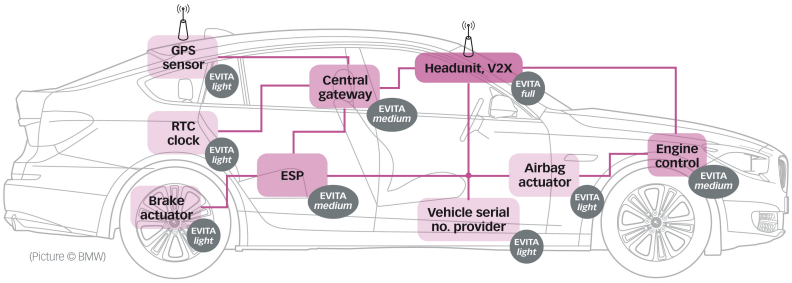

Fig. 1. Automotive on-board network architecture [11]

the security policy to be applied in a vehicle is the combination of an invariant policy for the usage control of cryptographic credentials of Electronic Control Units (ECU), and a flexible networking security policy. The credential usage control policy is enforced by the HSM and possibly through the virtualization of the ECUs if applications on the same ECU have to be segregated. In contrast, the networking security policy is enforced by all network elements. Moreover, the access control architecture must also allow enforcement of rules that limit the traffic on the buses under consideration, based on trusted authentication or other security mechanisms like traffic filtering or secure logging. However, as highlighted in the previous section, the enforcement of these different security mechanisms depends on a specific event or situation. For instance, while communicating with external entities like vehicle-to-infrastructure, it is preferable to apply the traffic filtering rules to limit the computation load on the HSM, which is responsible for the verification of cryptographic operations. Applying such rules will eventually increase the performance of on-board system. However, always applying such kind of rules is not desirable, as the enforcement of rules requires that the vehicle is in a specific context as well as a specific security event is active. To dynamically enforce these different sets of security policies, we call these policies as reaction policies [3]. In an on-board architecture, we need a system in which policy enforcement decisions are based on specific arguments in order to attain more fine grained enforcement of security policies.

3 Related Work

Based on the general idea of the platform in [8], the authors in [5] use the logical argumentation to support generation of the low-level rules from high-level policies. In [4] the authors treat the problem of resolving the possible anomalies in firewall policies using Argumentation for Logic Programming with Priorities (LPP). The use of this framework allows preferences to be encoded, thus allowing complex reasoning over the relative priorities between rules. The framework presented by Applebaum in [2] differentiates itself from these last two references through the introduction of the rationales behind each argument in the policy.

Applebaum propose to resolve conflictual situations in firewall policies by defining a potential ordering of the rationales behind each argument (rule). Firewall can then resolve anomalies and conflictual rules through this order of priorites. In [2] Applebaum defines a static order of priorities for the rationales behind the firewall rules. However, administrators can decide in specific cases to change the order of rationales priority. For instance, giving "allow legitimate senders" rationale a higher priority than "allow programs". In this case, firewall administrators are obliged to update the firewall configuration for each required change in order priorities. Argumentation has also received attention in the community of multi-agent systems in recent years, with a particular interest in the use of argumentative models from the informal logic viewpoint such as that by Walton and Krabbe [12,15]. Additionnaly, Bench-Capon introduces the value-based framework in [7], and he extends it in [10] to provide meta-arguments to reason about preference levels. However, current AL approaches target the system from the static viewpoint, while ignoring any dynamic change in the system state or security event, during system evolution.

4 Argumentative Logic-driven Reasoning Engine

In our approach, we design an argumentative framework allowing the automation of adapting priorities order, according to the current security situation and building the security metric. To achieve this goal, our approach can be summarized in the following three steps :

- For each security event, we define the rationale behind each possible countermeasure.
- According to the contextual values and depending on security events, we build the security metric which define the order between rationales.
- In case of conflict between countermeasures, we reason about the risk analysis to decide which countermeasure is more important to apply.

In this section, we will define the role of different parts involved in the architecture described in Figure 2 and the relations between them.

4.1 Security Policies

We start by defining the reaction policies, which presents the knowledge base of security policies defined by the target organization. The reasoning module performs the mapping between the collected information and security policies in order to identify the rules that match. Reaction security policies are presented as sets of rules in which we define the list of possible countermeasure for possible security events that may occur in system evolution. We present in Table 1 some examples of countermeasures for some security events according to automotive system case [1]. The functional experts are responsible for giving rationales for different countermeasure introduced in the security policy. These rationales are

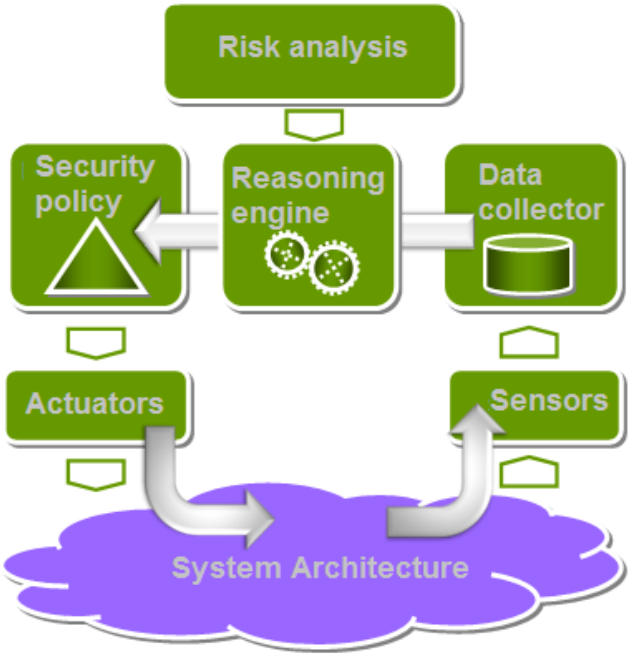

Fig. 2. Architecture of system detection & reaction

defined to express the result of the application of each countermeasure on the system. In the Policy Administration Point (PAP) part, we assign to each rule defined in the policy, the rationale behind it. For instance, we attribute "performance" as a rationale for the countermeasure "Reduce frequency of beaconing and other repeated messages" when "message saturation" threat occurs. According to the contextual values and depending on security events, we build in the PAP the security metric which defines the priorities order between rationales. According to the security and contextual information collected by the data collection part, the Policy Decision point (PDP) refers to the reasoning module to choose, among the rules introduced into security policies, the suitable rule to apply(e.g., if "performance" rationale has the highest priority in the context on which the "message saturation" threat was executed, the PDP choose "Reduce frequency of beaconing and other repeated messages"). Finally, the decision will be applied at the level of the Policy Enforcement Point (PEP) of the targeted system component. In complex system, the PEP is integrated in all the system components, to enforce the reasoning engine decision in the relevant component.

4.2 Reasoning Engine

In order to ensure system security, policy reactions must maintain a certain level of intelligence and dynamicity. Those properties allow security systems, accord-

Threats	**Countermeasure**
Message saturation	Reduce frequency of beaconing and other repeated messages
	Add source identification (IP address equivalent) in V2V messages
	Limit message traffic to V2I/I2V and implement station registration
Manipulation of relayed ITS messages en route	Plausibility checks on incoming messages
	Include a non-cryptographic checksum of the message in each message sent
	Remove requirements for message relay in the ITS BSA
Wormhole attacks	Use INS or existing dead-reckoning methods (with regular - but possibly infrequent - GNSS corrections) to provide positional data
	Implement differential monitoring on the GNSS system to identify unusual changes in position
	Use broadcast time (Universal Coordinated Time - UTC - or GNSS) to timestamp all messages

Table 1. Potential security countermeasures to threats in an ITS system [1]

ing to a specific security situation, to choose the best countermeasures to apply from the security metric. At the same time, the system enforces security conditions and a better efficiency of execution. Many approaches use argumentative logic (AL) [2] to provide rationales behind security policies. However, current AL approaches typically target the system from the static viewpoint and developed the metric of arguments, while ignoring any dynamic change in the system state or security event, during system evolution. To do this, security tools must have intelligent reasoning capabilities as the one of the human brain. To develop an approach for dynamic policy reactions, we inspire from the human way of arguing. Argumentation has been shown to be a useful framework for formalizing non-monotonic reasoning. It is a branch of logic that enables reasoning about the arguments to resolve inconsistencies in logic theory.

Argumentative logic

An argumentation system is a way for an agent to manage conflicting information and draw conclusions. In an abstract argumentation system, the basic information is a set of abstract arguments, which may for example represent a given proposal, and conflicts between arguments are represented by a binary relation on the set of arguments. For two arguments A and B, the meaning of attacks(A,B) is that A represents an attack on B. We also say that a set

of arguments S attacks an argument B if B is attacked by an argument in S. We rely on the Dung's [6] definition of an argumentation logic, which states that:

Definition 1:
An argumentation framework is a pair $AF = < AR, attacks >$
Where AR is a set of arguments and attacks is a binary relation on AR, i.e. $attacks \subseteq ARXAR$.

Dung introduces also the notion of acceptability of arguments. An accepted argument, in a set of arguments, is the one that every attack on it is rebutted by an accepted argument form the set.

Definition 2:
An argument $A \in AR$ is acceptable with respect to set of arguments S (acceptable(A,S)), if:
$(\forall x)((x \in AR)\&(attacks(x, A)) \rightarrow (\exists y)(y \in S)\&attacks(y, x)$.

In security policies, we consider that each attack on a rule implemented in the policy is rebutted by itself because attack relation is considered as symetric from reaction security point of view. Thus, each security rule implemented in security policy is accepted. To adapt the argumentation system to the security requirements, we consider security policies as arguments. In addition, we define a relation of attack between arguments (rules), two or more different possible countermeasures that can be applied for a specific security event. For instance, in the automotive case, three different countermeasures are presented according to the "Message saturation" security threat as shown in Table 1. Based on the above mentioned Definition 1, these countermeasures represent two arguments that attack each other because we are in a conflicting situation and we need to choose the suitable countermeasure among them. In our approach, we define the conflict between independent countermeasures according to a specified threat as following:

Definition 3:
$\forall A \in T, \forall X_1, X_2 \in C_A$
$conflict\_independent(X1, X2) \leftarrow countermeasure(X1) \wedge countermeasure(X2)$.
Where T present the set of all the threats and C_A is the set of possible countermeasures according to threat A.

Attack relation may also occur between two or more different arguments, according to different threats, that the system is unable to apply at the same time as described in section 5.1. We define the conflict between dependent countermeasures according to two threats as following:

Definition 4:
$\forall A, B \in T, \forall X_1 \in C_A$ and $\forall X_2 \in C_B$

$conflict\_dependent(X1, X2) \leftarrow countermeasure(X1) \wedge countermeasure(X2) \wedge dependent(X1, X2)$.
Where T present the set of all the threats, C_A is the set of possible countermeasures according to threat A, C_B is the set of possible countermeasures according to threat B, and $dependent(X1, X2)$ is the predicate that inform the system about the dependence between countermeasures.

4.3 Risk determination

The purpose of this risk assesment is to assess whether the threats or security vulnerabilities are relevant according to the security level specified by the security goals. In our approach, we estimate the security risks based on the relevant threats, their likelihood/impact [13] that the threats will materialize as real attacks, any potential consequences on the system assets or possible severity of an attack for the stakeholders, and the resulting impact of that adverse event on the organization. We also consider the weakness that may occur when we apply a specific countermeasure. In the case of a complex attack, we are always facing conflicting situations where applying a countermeasure for a security event can deactivate the application of other countermeasures. In our approach, we are interested in the likelihood and the impact of each security threat. The likelihood depends on five factors that we affect range and value as shown in Table 2 (e.g., an attack that affect a standard equipment is more likely to occur than an attack that affect a specialized one, attackers can be better familiarized with standard than specialized equipment). The "likelihood" value is calculated from the sum of the five factors values. The threat group likelihood is evaluated through the sum obtained, likelihood increases with the decrease of the sum. For instance, for a sum belonging to the interval [0,3] the likelihood value is considered the most important "likely".

4.4 System Architecture

The system architecture shows the composition of the system in terms of components and interconnections between them. Through sensors and intrusion detectors installed in the various components of the system, data collection part is able to acquire security information as attack notifications and contextual data. Each component in the system architecture is linked by a sensor to detect all malicious activities that may occur, and actuators to apply different actions taken by the PDP. From the automotive viewpoint, the system architecture is composed of several components interlinked. The components of automotive system architecture are equipped by a hardware security module (HSM) that provides means to protect the plateform by protecting critical assets of the architecture. Once the reasoning module has taken the right decision to respond to an attack, the PEP is responsible for applying the countermeasure taken at the relevant component in the architecture. The information in complex system is distributed in all the system. Thus, in order to have a global view of all the security data,

Threat Group	Attack				Impact
	Factor	Range	Value	Likelihood	
Acquisition of personal information	Time	< = 1 day	0	2 (Possible)	3 (Medium)
	Expertise	Proficient	2		
	Knowledge	Restricted	1		
	Opportunity	Moderate	4		
	Equipment	Standard	0		
Acquisition of behavioural details	Time	< = 1 day	0	1 (Possible)	2 (Medium)
	Expertise	Proficient	2		
	Knowledge	Restricted	1		
	Opportunity	Difficult	12		
	Equipment	Specialized	3		
Denial of transmission	Time	< = 1 day	0	3 (Likely)	1 (Low)
	Expertise	Layman	0		
	Knowledge	Public	0		
	Opportunity	Easy	1		
	Equipment	Standard	0		
Denial of receipt	Time	< = 1 day	0	3 (Likely)	2 (Low)
	Expertise	Layman	0		
	Knowledge	Public	0		
	Opportunity	Easy	1		
	Equipment	Standard	0		

Table 2. Risk determination in an ITS system [13]

we need a data collector in our architecture which collects information from all components sensors.

4.5 Data Collection

Data collection is the process of combining and associating information regarding one or more entities considered in a knowledge framework. The aim of data collector is to improve capability for detection, identification and characterization of that entity. In modern decision support systems, information coming from several sensors is fused in order to overcome the uncertainty in a case. The main purpose of collection is to provide an overall picture of the significance of the information collected by different platforms to classify/identify the target entities and to have a new data set containing the meaningful data. Data collection part notifies the reasoning module when security events occur (individual or complex attacks) and the contextual information on which they were produced.

Security events

Security events present all the types of attacks that may occur on a system and harm its evolution in safety conditions. Nowdays attacks are becoming more and more complex which complicate the task of security tools to prevent them. And it becomes even more difficult in the case of coordinated attacks. A coordinated attack is the collaboration of several attacking sources to achieve a common goal. In order to achieve this goal, attacking sources, controlled by one or several attacking entities, may cooperate by resource sharing, task allocation, synchronization,etc. Many works focus on complex attack and the way to design them as in [14] where the authors define a formal description of individual and

coordinated attacks.

Contextual information

Data collector collects contextual information describing the circumstances in which the attacks occurred. Those information are extracted by the reasoning module that used it to reason and take the decision that best fits the context. From the automotive point of view, we distinguish in this paper three types of context :

- In-car
- V2V (Vehicule to Vehicule)
- V2I/I2V (Vehicule to Infrastructure / Infrastructure to Vehicle)

5 Dynamic Deployment of Security Policies

We present in this section the deployment of our approach, described in the last section. The security data (i.e., security event and contextual information) are collected, by the data collector, through the sensors integrated at different layer of system architecture. These data are extracted by the reasoning engine and compared by the security rules defined in the security policies, to identify the possible countermeasures to apply.

5.1 Evaluation of countermeasures

In some cases, the PDP can make the decision to apply two or more countermeasures (especially in the case of complex attack) where one of them prevents the execution of another countermeasure, or reduces the efficiency of its application as described in section 5.4. Each countermeasure taken by the system, may have one or several weakness points that can affect the reaction process and the maintenance of the system in security conditions. We present in Table 3 some of the weaknesses of the potential countermeasures that may degrade the overall security and safety of the system. Complex systems such as automotive on-board system arhictecture are more vulnerable to coordinated and complex attacks, due to the complexity of its components and the distributed nature of the system. The attackers try to damage the system through the intrusion to one or many components of the system.

5.2 An example of complex attack in automotive system

The coordinated attack described below requires a minimum number of Group of Coordinating Attackers (GCA), and it inflicts damage to automotive ITS system by executing two different attacks (Message Saturation (MS) and Manipulation of Relayed ITS messages in route (MR)): A part of attackers saturates the ITS server by sending a huge number of requests at the same time. While the ITS

Security Countermeasure	**Limitations**
Reduce frequency of beaconing and other repeated messages	Safety-critical messages may not be received quickly enough by affected vehicles
Add source identification (IP address equivalent) in V2V messages	The desired principles of anonymity within ITS are breached
	May not be available in the existing stack
Limit message traffic to V2I/I2V when infrastructure is available and implement message flow control and station registration	The coverage of the ITS infrastructure would have to be extensive
	The speed of response to an incident would deteriorate (however, response times would be deterministic)
	Current IEEE 802.11 technologies do not support flow control

Table 3. Security countermeasures and their limitations [1]

system is unable to process all the incoming messages, the other part of attackers captures those messages as an ITS server and manipulates them. We use the modelisation of complex attacks described in [14] to define an example of complex attack in automotive system (CXA) corresponding to the type "Coordinated Attack with Load Accumulation – CALA". CALA is defined in [14] as a coordinated attack in which attackers accumulate their capabilities. This offers execution of the attack in a distributed and simultaneous way.

(CALA): CALA_ ComplexAttack (GCA_{CXA}, ITS)
$\min_{CXA} = 17$
$A_{CXA} = \{network\_access(attackerID, ITS)\}$; $A'_{CXA} = \{\}$
$B_{CXA} = \{knows(attackerID, is\_on(ITS))\}$; $B'_{CXA} = \{MS(ITS)))\}$
$\Gamma_{CXA} = \{is\_on(ITS, MS(ITS))\}$; $\Gamma'_{CXA} = \{MS(ITS), MR(ITS)\}$

A_{CXA}, B_{CXA} and Γ_{CXA} present the precondition predicates that have to be satisfied to allow the action execution. A'_{CXA}, B'_{CXA} and Γ'_{CXA} present the post-condition predicates that become true after the action execution. The $\min_{CXA}$ value is obtained from the sum of the opportunity values of the threats involved in the complex attack. Automotive experts define the opportunity value for attacks as the number of attackers required to achieve the attack. In this case, we have two different attacks (MS and MR) with the opportunity values of each one is 12 and 5, the resulting $\min_{CXA}$ value is 17.

5.3 System reaction and inconsistency resolution

To mitigate this complex attack, we apply the approach defined in section 4:

- We start by defining, for each security event, the rationale behind each possible countermeasure as presented in Table 4, where we have defined the

priorities for "Message saturation" threat, we start with the "in-car" context then "V2V" context, and finally "V2I/I2V" context:

Context	Rule	Countermeasure	Reason	Attack
In-car	1	Reduce frequency of beaconing and other repeated messages	Performance	2,3
	2	Add source identification (IP address equivalent) in V2V messages	Data-source authenticity	1,3
	3	Limit message traffic to V2I/I2V and implement station registration	Filtering	1,2
V2V	2	Add source identification (IP address equivalent) in V2V messages	Data-source authenticity	1,3
	1	Reduce frequency of beaconing and other repeated messages	Performance	2,3
	3	Limit message traffic to V2I/I2V and implement station registration	Filtering	1,2
V2I/I2V	3	Limit message traffic to V2I/I2V and implement station registration	Filtering	1,2
	2	Add source identification (IP address equivalent) in V2V messages	Data-source authenticity	1,3
	1	Reduce frequency of beaconing and other repeated messages	Performance	2,3

Table 4. Priority order between reasons for message saturation threat

These requirements are defined in a formal way as following:
$rule1 : countermeasure(reduce\_frequency) \leftarrow threat(message\_saturation)$
$rule2 : countermeasure(add\_source) \leftarrow threat(message\_saturation)$
$rule3 : countermeasure(limit\_traffic) \leftarrow threat(message\_saturation)$

We include the incompatibility predicate defined in Definition 3 to ensure that different countermeasure cannot be part of the same acceptable argument:

$conflict\_independent(countermeasure(reduce\_frequency),$
$countermeasure(add\_source)) \wedge conflict\_independent(countermeasure$
$(limit\_traffic), countermeasure(reduce\_frequency)) \wedge conflict\_$
$independent(countermeasure(limit\_traffic), countermeasure(add\_source))$.

This conflict is resolved through the priority predicate which define the rationale priority order:

$priority(1, rule1, incar, performance)$:
$rule1 \leftarrow threat(message\_saturation) \wedge context(incar)$
$priority(2, rule2, incar, authenticity)$:

$rule2 \leftarrow threat(message\_saturation) \wedge context(incar)$
$priority(3, rule3, incar, filtering) :$
$rule3 \leftarrow threat(message\_saturation) \wedge context(incar)$

- According to the complex attack (CXA), we present in the Table 5, the best countermeasure to apply corresponding to the reason having the higher priority, and depending to the context in which the security event was produced (In-car).

Security event	Countermeasure	Reason	Weakness	Likelihood / impact
MS	Reduce frequency of beaconing and other repeated messages	Performance	Safety-critical messages may not be received quickly enough by affected vehicles	(1,3)
MR	Include a non-cryptographic checksum of the message in each message sent	Availability	A subverted legitimate ITS-S possess all of the necessary algorithms to compute a valid checksum for a maliciously modified message	(2,3)

Table 5. Default countermeasures configuration for complex attack

When the system applies the "Reduce frequency of beaconing and other repeated messages" countermeasure, the safety-critical messages may not be received quickly enough by affected vehicles. Thus, the system is not able to include a non-cryptographic checksum of the message in each message sent, which is the countermeasure for "Manipulation of relayed ITS messages in route" threat. The weakness of the "Message saturation" countermeasure affects the treatment of the second security event which can be detected through conflict_dependent predicate defined in Definition 4:

$conflict\_dependent(countermeasure(reduce\_frequency), countermeasure(checksum))$

To resolve this conflict, the reasoning engine refers to the risk analysis part to identify which security threat has the greatest likelihood/impact value.

$prefer(threat(manipulation\_ITS\_messages), threat(message\_saturation) \leftarrow like\_impact(manipulation\_ITS\_messages) > like\_impact(message\_saturation)$

The prefer predicate inform the reasoning engine about the most important threat from the threats having conflictual countermeasures. The processing of

the “Manipulation of relayed ITS messages in route” is more important than the “Message saturation” thread because of the likelihood/impact values. We are confronted here to a conflict between two countermeasures. Applying the best countermeasure for each security event is impossible, that is why we refer to Table 4 and we pass to the second possible countermeasure for “Message saturation” which ensures Data-source authenticity. We present in Table 6 the configuration of countermeasures adopted by the reasoning engine to mitigate the complex attack.

Security event	Countermeasure	Reason	Weakness	Likelihood / impact
MS	Add source identification (IP address equivalent) in V2V messages	Data-source authenticity	The desired principles of anonymity within ITS are breached	(1,3)
MR	Include a non-cryptographic checksum of the message in each message sent	Availability	A subverted legitimate ITS-S possess all of the necessary algorithms to compute a valid checksum for a maliciously modified message	(2,3)

Table 6. Countermeasures adopted for complex attack

As we can see in Table 6, the application of each countermeasure of MS and MR does not affect the execution of the other one. We consider this configuration as the best that system can apply to maintain the system in safe conditions and offer better performance.

6 Conclusion and Future Work

In order to protect system from modern attacks, it is necessary to have a dynamic and intelligent enforcement of security policies. We consider the argumentative logic driven system the most appropriate to achieve this objective. In this work we showed how to improve the existing argumentation framework, we proposed a new approach that allows us to consider the context of security situations in order to take the suitable and dynamic decisions. We are currently working to include the case of dependancies between system components and other functional and non functional requirements of the system. More specifically, we are designing an approach that consider the system dependancies in reasoning while considering other requirements such as performance, cost, etc.

References

1. Etsi tr 102 638: ”intelligent transport systems (its); vehicular communications; basic set of applications; definitions”.

2. A. Applebaum, K. N. Levitt, J. Rowe, and S. Parsons. Arguing about firewall policy. In B. Verheij, S. Szeider, and S. Woltran, editors, *COMMA*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 91–102. IOS Press, 2012.
3. F. Autrel, N. Cuppens-Boulahia, and F. Cuppens. Reaction policy model based on dynamic organizations and threat context. In E. Gudes and J. Vaidya, editors, *DBSec*, volume 5645 of *Lecture Notes in Computer Science*, pages 49–64. Springer, 2009.
4. A. K. Bandara, A. C. Kakas, E. C. Lupu, and A. Russo. Using argumentation logic for firewall policy specification and analysis. In R. State, S. van der Meer, D. O'Sullivan, and T. Pfeifer, editors, *DSOM*, volume 4269 of *Lecture Notes in Computer Science*, pages 185–196. Springer, 2006.
5. A. K. Bandara, A. C. Kakas, E. C. Lupu, and A. Russo. Using argumentation logic for firewall configuration management. In *Integrated Network Management*, pages 180–187. IEEE, 2009.
6. H. Bar-El. Intra-Vehicle Information Security Framework. In *Proceedings of the 7th escar Conference*, Düsseldorf, Germany, 2009.
7. T. J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *J. Log. Comput.*, 13(3):429–448, 2003.
8. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.
9. K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 447 –462, May 2010.
10. S. Modgil and T. J. M. Bench-Capon. Integrating object and meta-level value based argumentation. In P. Besnard, S. Doutre, and A. Hunter, editors, *COMMA*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 240–251. IOS Press, 2008.
11. E. Project. E-safety Vehicle InTrusion protected Applications. `http://www.evita-project.org`.
12. C. Reed. Dialogue frames in agent communication. In Y. Demazeau, editor, *ICMAS*, pages 246–253. IEEE Computer Society, 1998.
13. A. Ruddle, D.Ward, B. Weyl, M. S. Idrees, Y.Roudier, M.Friedewald, T.Leimbach, A.Fuchs, S.Gürgens, O.Henniger, R. Rieke, M.Ritscher, H.Broberg, L. Apvrille, R. Pacalet, and G.Pedroza. Security Requirements for Automotive On-Board Networks based on Dark-side Scenarios. Technical Report 2.3, EVITA Project, 2010.
14. L. Samarji, F. Cuppens, N. Cuppens-Boulahia, W. Kanoun, and S. Dubus. Situation calculus and graph based defensive modeling of simultaneous attacks. In G. Wang, I. Ray, D. Feng, and M. Rajarajan, editors, *CSS*, volume 8300 of *Lecture Notes in Computer Science*, pages 132–150. Springer, 2013.
15. D. Walton and E. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. G - Reference,Information and Interdisciplinary Subjects Series. State University of New York Press, 1995.